big-data-report.pdf

by Aditya Kulkarni

Submission date: 04-Aug-2024 02:34PM (UTC-0500)
Submission ID: 2427080188
File name: big-data-report.pdf (414.21K)
Word count: 4608
Character count: 26191

Real-Time News Analysis using Deep Learning and Large Language Models

Mr. Aditya Kulkarni
Department of Computer Science
University of Texas at Dallas
Dallas, United States of America
axk230069@utdallas.edu

Mr. Harshal Agrawal
Department of Computer Science
University of Texas at Dallas
Dallas, United States of America
hxa230036@utdallas.edu

Mr. Kamal Kaushik Varanasi
Department of Computer Science
University of Texas at Dallas
Dallas, United States of America
kxv230005@utdallas.edu

***Abstract*— In recent years, the polarization of news outlets has intensified, raising significant concerns about the impact of biased political reporting on public opinion and democratic processes. As news is the primary and most direct source of information about ongoing events, the introduction of unjust subjectivity can severely undermine democratic ideals and distort public perception. Moreover, the advent of technology has led to a massive increase in the volume of news articles published daily, accessible through a variety of digital platforms. This proliferation has made it challenging for traditional methods to analyze such vast data in real-time. Consequently, there is a pressing need for automated real-time data processing solutions for detecting and analyzing political news bias. This paper presents a study on the detection of political bias, sentimental analysis and news categorization in real-time. We developed a Spark Streaming application that continuously processes raw news data from real-time sources. Our system analyzes news bias and sentiment and categorizes articles using deep learning techniques, while the results are visualized using Kibana. We leverage Kafka for data ingestion and output, and utilize NewsAPI and the Reddit API as our real-time data sources. By leveraging Kafka for real-time streaming and Deep Learning as well as Large Language models for classification**

Keywords—Media bias detection, Sentiment Analysis, BERT, Spark Streaming, ELK Stack, web scraping

I. Introduction

In today's digital age, the swift expansion of online news has significantly altered the manner in which information is distributed and consumed. This transformation has facilitated broader access to a variety of viewpoints; however, it has also raised significant concerns regarding the existence and influence of bias in political news coverage. Media bias, whether deliberate or unintentional, possesses the potential to shape public perceptions, sway political beliefs, and ultimately impact democratic processes. Consequently, the identification and examination of bias within news content have emerged as vital areas of inquiry for researchers, journalists, and policymakers alike.

Political news bias can present itself in numerous ways, including selective reporting, framing techniques, and the use of loaded language, all of which can skew the representation of facts and events. Conventional approaches to detecting bias have typically depended on qualitative analyses conducted by experts, which, although informative, can be labor-intensive and prone to human error. With the introduction of natural language processing (NLP) and machine learning (ML), there is an expanding opportunity to automate and improve the identification of bias in news articles.

The emergence of technology has led to the daily publication of an extensive array of news articles, which are readily available across various digital platforms. This surge in news content poses significant challenges for traditional methods tasked with analyzing such huge volume of data in real time. Consequently, there exists an urgent need for automated solutions capable of processing data in real time to identify and analyze bias in political news. In response to these challenges, we have developed a Spark Streaming application that continuously processes raw news data from real-time sources. Our system uses deep learning algorithms to categorize news articles and evaluate news bias and sentiment, before showcasing our findings through an interactive visualization dashboard built using the ELK stack. We utilize Kafka for data ingestion and output, integrating NewsAPI and the Reddit API as our sources for real-time news data. This comprehensive architecture facilitates the effective management and analysis of substantial volumes of news content, thereby providing prompt insights into media bias trends.

The remainder of this paper is structured as follows: Section 2 review the background and the related literature on media bias and computational detection methods. Sections 3 and 4 outlines our methodology, including data collection, evaluation metrics, and model development. Finally, Section 6 discusses the implications of our findings and suggests directions for future research.

II. Literature Review

In this section we are going to review some of the approaches employed by researchers in detecting the bias and sentiment in news articles.

A. Media Bias Detection

Detecting media bias is a relatively new research area in the field of natural language processing. The existing methodologies can be braodly classified into two categories:

1) Traditional ML Based Approaches

These methodologies primarily relied on manually designed features, which included lexical characteristics associated with word selection and usage [9, 10], morphosyntactic attributes concerning grammar and sentence construction, and semantic aspects that pertained to the meaning of the text [11]. The emphasis was on extracting the appropriate features during feature selection. The authors employed classical machine learning algorithms like naive bayes, logistic regression and SVM to detect media bias based on the extracted features.

2) Deep Learning Based Approaches

Recent advancements in deep learning have considerably enhanced the detection of media bias. Deep learning architectures, particularly Recurrent Neural Networks (RNNs) and their derivatives such as Long Short-Term Memory (LSTM) networks, excel at understanding sequential sentence structures, thereby facilitating more precise identification of bias. Although traditional RNNs face challenges with long-term dependencies, LSTMs effectively address this issue through their internal memory capabilities, which significantly improves the accuracy of bias detection.

Transformers, which utilize self-attention mechanisms, have surpassed RNNs in various applications, including media bias detection. Models based on BERT, as demonstrated by researchers such as Baly et al. (2020) [12] and Chen et al. [13], have produced results that exceed those of traditional models. These advancements underscore the significance of high-level features and discourse structures in bias identification. Additionally, methodologies like the Gaussian Mixture Model and Multi-task Learning contribute to further improvements in detection accuracy. Recent studies conducted by Krieger et al. [14] and others have introduced transformer-based models specifically designed for media bias detection, showcasing exceptional performance. The emphasis on sentence-level analysis and the application of subframe indicators illustrate the complexity and versatility of modern models in the identification of media bias.

B. Sentiment Analysis

Researchers have always been curious in identifying sentiment in text, right from the early years of NLP research. Earliest sentiment analysis approaches were rule-based dependent on lexical features such as words annotated with sentiment polarity scores. Features like n-grams, term frequencies and POS tags have actively been used for feature extraction along with traditional machine learning algorithms like naive bayes, random forest and SVM for classification. Advancements in deep learning research led to researchers preferring LSTMs and CNNs over lexical and syntactic features to better learn contextual features, leading to a boost in performance. More recently, pre-trained models like BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer) have been adapted for sentiment analysis, achieving state-of-the-art results due to their ability to capture context and semantics effectively.

The research on sentiment analysis of news articles encompasses a variety of approaches with different end goals. Majority of these approaches depend on lexical based features for analyzing the sentiment of a news article. Reis et al. [15] conducted a comprehensive study of 69,907 headlines from four major media outlets, revealing that sentiment polarity significantly influences news article popularity. Their findings showed that both negative and positive headlines tend to attract more attention than neutral ones. While, the authors of Godbole et al. [16] developed an algorithm that utilizes sentiment lexicons to analyze sentiment words and entities in news and blogs. Their approach focused on polarity and subjectivity across various domains, including health and politics

Sentiment analysis of real-time news data generally includes reading data from real-time sources, data preprocessing, feature extraction and classification. Imran et al. [17] introduced StreamSensing, a sentiment analysis approach leveraging Apache Spark for efficient processing of real-time, unstructured, and noisy data streams. The method was applied to lexicon data to identify notable trends, which could be generalized to any real-time text stream.

III. Methodology

To collect real-time news, we employ the NewsAPI and RedditAPI to extract news articles from the r/news subreddit. Due to a limit of 100 requests on NewsAPI, we fetch first few articles from NewsAPI and then switch to streaming the r/news subreddit. The Reddit API (Application Programming Interface) allows developers to interact with Reddit's data programmatically. It provides endpoints for accessing various data on Reddit, including posts, comments, user information, and subreddit details. By leveraging this API, we can automate the process of collecting and analyzing news articles.

We set up a Reddit API client using the ***PRAW*** (Python Reddit API Wrapper) library to pull new posts as soon as they are posted on the subreddit. PRAW simplifies the interaction with the Reddit API and allows us to write concise and readable code. Each post is parsed to extract relevant information like content, source, headline, etc. [3]

The extracted data is then sent to a Kafka topic for real-time streaming. **Apache Kafka** is a distributed event streaming platform capable of handling trillions of events a day. It is designed to provide high throughput and low latency for real-time data feeds. Kafka's core components include producers, topics, brokers and consumers [4]. Producers are responsible for sending data to Kafka topics, while consumers read data from these topics. Topics are partitioned logs where data is stored, and brokers are Kafka servers that handle data storage and retrieval. [4]

We chose Apache Kafka for its robustness and scalability, which makes it suitable for real-time data streaming and integration with various data processing and analysis tools [4].

Here is a brief overview of how we used Apache Kafka in our project:

1. The news articles streamed from Reddit API is parsed and sent to a kafka topic "news" in JSON format where key is the source of news and value is the news headline.
2. The Pyspark code acts as a consumer of this topic and subscribes it. The received news is processed and the data is published to another topic in JSON format with headline, sentiment, bias_rating, category, and named-entity counts as keys.
3. This second topic is subscribed by Logstash and the data is parsed and passed further to Elasticsearch and Kibana for further processing.

The incoming news data from Kafka is processed and labelled using three deep learning models, each for the tasks described before (bias detection, sentiment analysis, and news categorization). We describe the training datasets and the workings of the models for all the three tasks below:

C. Datasets and Models Description

1) Bias Detector

For this task, we trained a deep learning model to detect political bias in news articles, leveraging Spark NLP and BERT embeddings [1]. We scraped the dataset from AllSides.com using ***selenium*** and google chrome *web driver* on Python. It included articles labeled with their respective political biases, such as **left**, **center**, and **right**. We applied standard preprocessing steps to the dataset, including removing stop-words, tokenization, and generating sentence and word embeddings using the *small_bert_L8_512* model [2]. BERT embeddings were chosen for their ability to capture the context of words within sentences, providing a richer text representation. After the embedding layer, the architecture includes multiple neural network layers. These include recurrent layers, convolution and pooling layers, and fully connected layers.

The model's architecture is based on the Spark NLP's `***ClassifierDLApproach***`, which is a deep learning-based classifier designed for text classification tasks. We incorporated this classifier into a Spark ML pipeline, allowing for streamlined training and evaluation [1]. Key hyperparameters, such as learning rate, number of epochs, and batch size, were meticulously tweaked to optimize the model's performance. The training process involved a 90-10 split of the dataset into training and testing sets, with the model trained over 100 epochs, using a learning rate of 0.001 and a batch size of 1. These parameters were selected after extensive experimentation to balance training duration and model performance.

The model was evaluated using metrics such as accuracy, precision, recall, and F1-score. On the test set, the model achieved an accuracy of 74%, indicating a reasonably good performance in detecting political bias in news articles. Despite this success, we encountered several challenges. One significant challenge was the imbalance in the dataset, with certain political biases being underrepresented. Additionally, the subtle and context-dependent nature of political bias posed difficulties in capturing it effectively through simple preprocessing steps.

To address these challenges, fwe plan to employ data augmentation techniques to create a more balanced dataset, and advanced embeddings such as *RoBERTa* or GPT to potentially enhance the overall performance. Furthermore, implementing ensemble methods to combine multiple models might improve accuracy and robustness of the model.

2) Sentiment Detector

In addition to the political bias detection model, this paper also presents a sentiment analysis model trained on "News Sentiment Analysis" dataset from Kaggle. The dataset contains text data labeled with three sentiment categories, namely **positive**, **negative**, and **neutral**. The dataset consists of 10,440 news headlines, out of which 6008 records are of neutral sentiment, 3215 are positive and 1464 are negative. Similar to the previous model, we applied standard preprocessing steps, including removing stop-words, tokenization, and generating embeddings using the *small_bert_L8_512* [2] model. The sentiment analysis model utilized the same core architecture as the political bias detection model, built around Spark NLP's ***ClassifierDLApproach*** and integrated into a Spark ML pipeline. [1]

The model achieved an accuracy of 80% on the test set, demonstrating its effectiveness in classifying sentiment in text data. This performance highlights the robustness of the *ClassifierDLApproach* combined with BERT embeddings in handling various text classification tasks. While the sentiment analysis task shared many preprocessing and architectural elements with the political bias detection model, it specifically focused on capturing the overall sentiment expressed in the text.

Overall, the sentiment analysis model underscores the versatility and capability of using deep learning techniques and BERT embeddings for different text classification applications. The achieved accuracy of 80% provides a strong foundation for further enhancements and confirms the potential of this approach in sentiment analysis.

3) News Category Detector

For the news category detection task, a dataset [5] from the spark-nlp workshop was used. This dataset includes four categories: **World**, **Sci/Tech**, **Business**, and **Sports**, each represented equally. It consists of 120,000 training samples and 7,600 testing samples.

Unlike the case of previous tasks, we fine-tuned the DistilBERT model from Hugging Face [6] for this task. ***DistilBERT*** [7] is a smaller, faster, and more cost-effective version of the original BERT model, retaining 97% of BERT's understanding while being 60% faster. During finetuning, a new classification head, which is a fully connected dense layer, is added on top of the pretrained model. The output layer is then adjusted to match the number of labels, which in this case is four, resulting in an output layer with four units.

To fine-tune the *DistilBERT* model, the *transformers* library in Python was used. For preprocessing, the text in the dataset was tokenized using the base model *distil-bert-uncased*. The model was fine-tuned in a distributed fashion for 3 epochs with a batch size of 8. An accuracy of 88% was achieved in the first epoch, improving to 89% by the final run. Additional evaluation metrics are detailed in the next section.

This model demonstrates the benefits of using a pre-trained model that has been trained on a large corpus of data. The accuracy distribution during the training period indicates that this approach can achieve acceptable accuracy without requiring a vast amount of data. Future work will explore more innovative methods for training models and leveraging pre-trained models to our advantage.

D. Dashboard

Now that the models are trained and ready to be used for real time prediction, the labelled data is then stored and visualized using the **ELK Stack**. The ELK Stack is a powerful set of tools provided by Elastic for searching, analyzing, and visualizing data in real-time. ELK stands for Elasticsearch, Logstash, and Kibana:

1. **Elasticsearch**: Elasticsearch is a distributed, RESTful search and analytics engine capable of handling a wide variety of data types including textual, numerical, geospatial, structured, and unstructured data. It stores the data and provides search and analysis capabilities [8].
2. **Logstash**: Logstash is a data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch. It allows us to process and enrich the collected data before storing it in Elasticsearch [8].
3. **Kibana**: Kibana is a data visualization and exploration tool used for visualizing Elasticsearch data and navigating the Elastic Stack. It provides a user-friendly interface for creating dashboards, graphs, and charts to visualize and analyze the data. The Kibana dashboard has multiple insightful visualizations which are mentioned in the Results section. [8]

IV. RESULTS

A. Model evaluation: Sentiment Detector

	precision	recall	F1	support
negative	*0.55*	*0.59*	*0.57*	*150*
neutral	*0.86*	*0.84*	*0.85*	*564*
positive	*0.81*	*0.82*	*0.82*	*330*
accuracy			*0.80*	*1044*

The model's performance metrics for sentiment analysis reveal a balanced but varied performance across different sentiment classes. The overall accuracy is 80%, indicating that the model correctly predicts the sentiment for 80% of the test samples. Breaking down by class, the model performs exceptionally well on the 'neutral' sentiment, achieving an F1-score of 0.85 with high precision (0.86) and recall (0.84). For 'positive' sentiment, the performance is also robust, with an F1-score of 0.82, precision of 0.81, and recall of 0.82. However, the 'negative' sentiment class shows a relatively lower performance, with an F1-score of 0.57, precision of 0.55, and recall of 0.59. This discrepancy indicates that while the model is strong in predicting 'neutral' and 'positive' sentiments, it struggles more with accurately identifying 'negative' sentiments. The macro average F1-score of 0.75 reflects this imbalance, suggesting potential areas for improvement, particularly in enhancing the model's ability to detect negative sentiments. This model is trained with the following parameters: Learning rate = 0.01 for 100 epochs with batch size=8.

B. Model evaluation: Bias Detector

	precision	recall	F1	support
center	*0.68*	*0.60*	*0.64*	*901*
left	*0.64*	*0.75*	*0.69*	*1172*
right	*0.80*	*0.71*	*0.75*	*944*
accuracy			*0.74*	*3017*

The performance metrics for the political bias detection model reveal that it has a moderate overall accuracy of 74%, indicating that the model correctly predicts the bias for 74% of the test samples. Analyzing by class, the model shows varying degrees of performance.

The 'leaning right' bias class has the highest precision (0.80) and a solid F1-score (0.75), suggesting the model is quite effective at identifying right-leaning biases, albeit with a slightly lower recall (0.71). The 'leaning left' class, although having a lower precision (0.64), demonstrates a higher recall (0.75), resulting in a reasonably balanced F1-score of 0.69. The 'center' bias class exhibits the lowest performance, with a precision of 0.68, recall of 0.60, and an F1-score of 0.64, indicating that the model struggles the most with accurately classifying centrist biases. The macro average F1-score of 0.69 reflects this variability across classes, while the weighted average F1-score of 0.70 takes into account the class distribution, highlighting the model's overall performance. These results suggest that while the model performs adequately, there is room for improvement, particularly in better distinguishing centrist and left-leaning biases. This model is trained with the following parameters: Learning rate = 0.01 for 100 epochs with batch size=8.

C. Model evaluation: Category Detector

	precision	recall	F1	support
Business	*0.80*	*0.82*	*0.81*	*1611*
Sci/Tech	*0.92*	*0.98*	*0.90*	*1930*
Sports	*0.94*	*0.96*	*0.95*	*2204*
World	*0.87*	*0.86*	*0.85*	*1855*
accuracy			*0.89*	*7600*

The performance metrics for the news category detection model reveals a high accuracy score, i.e. 89% along with high precision and recall scores in almost all labels. The 'Sports' label has the best scores in case of almost all the metrics (precision score = 0.94, recall score = 0.96 and F1 score = 0.95). Followed with 'Sci/Tech', the precision reduced to 0.92, but has the best recall score, 0.98 among all the labels. The F1 score being 0.9. 'World' label has precision as 0.87, recall 0.86 and an F1 score of 0.85. 'Business' label relatively performs the worst but still retaining a score of 0.8 precision and 0.83 recall and 0.81 F1 score. 'Business' label has the potential for improvement, in turn improving the overall model's ability to perform.

D. Dashboard

The complete pipeline from reddit API to Kibana dashboard processes data at an average rate of **40 seconds per 100 articles**. Depending upon the activity in the subreddit being streamed, we observed that during peak hours and major news announcements, about 10-20 articles are posted every hour.

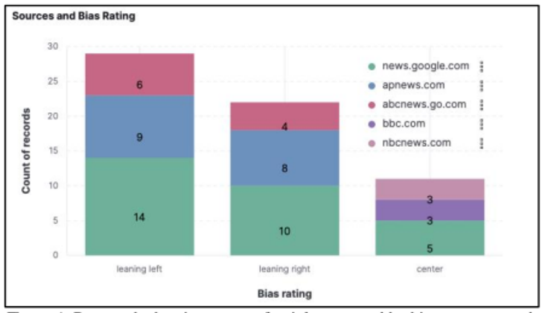

Figure 1. Bar graph showing count of articles grouped by bias category and top 3 sources for each category.

Figure 1 shows the distribution of news sources across different bias ratings, with a clear concentration of sources leaning left. While there are some sources leaning right and in the center, they are significantly outnumbered by those leaning left.

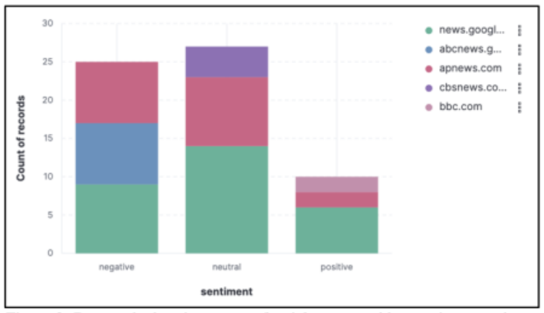

Figure 2. Bar graph showing count of articles grouped by sentiment and top 3 sources for each sentiment.

Figure 2 reveals a notable concentration of articles in the neutral sentiment range for all outlets, with a smaller but still significant number of articles exhibiting negative sentiment. Positive sentiment articles are notably less frequent across all sources. It's important to note that this chart only provides a snapshot of sentiment and doesn't delve into the specific content or context of the articles.

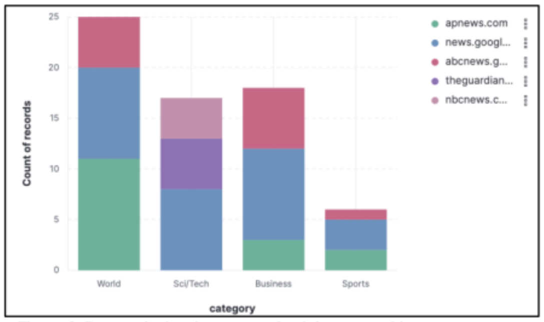

Figure 3. Bar graph showing count of articles grouped by news category and top 3 sources for each category.

Figure 3 shows the number of articles in different categories for five news outlets: apnews.com, news.google.com, abcnews.go.com, theguardian.com, and nbcnews.com.The chart shows that apnews.com has the most articles in all categories, followed by news.google.com. Theguardian.com has the fewest articles in all categories.

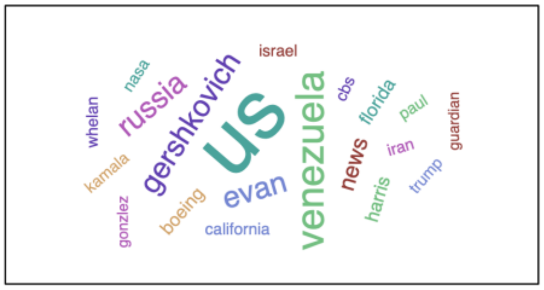

Figure 4. Word cloud showing the most frequently mentioned terms in headlines.

Figure 4 shows the most frequent terms used across all the news articles. This reflects the most recent buzzwords in the headlines. We can observe that most of the terms are from global("world") politics as that is the most common news category analyzed as shown in the Figure 5.

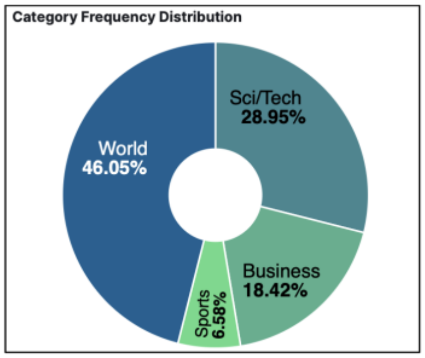

Figure 5. Pie chart for category distribution

Figure 6 shows percentage distribution of the sentiments of news analyzed. And, most of the news are categorized as negative which maybe due to the ongoing global conflicts.

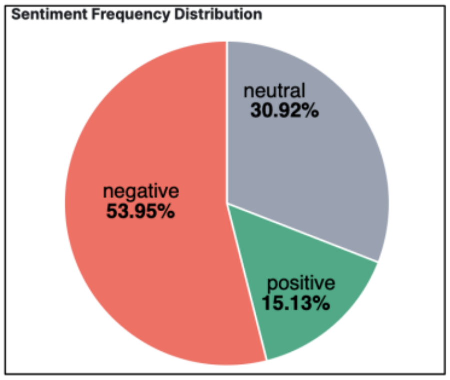

Figure 6. Pie chart for sentiment distribution

V. Conclusion and Future Work

In conclusion, the aim of this paper was to incorporate the big data technologies of today's world and utilize it to process and analyze streams of data in real-time. Our findings underline the importance of NLP techniques in media analytics, demonstrating how these tools can help identif and avoid bias, as well as negetively, in news articles or sources. This encourages users to seek diverse perspectives, promoting a more balanced and informed consumption of news.

By leveraging Spark's distributed capabilities, training deep learning models on large datasets in a parallelized fashion, significantly reduced the training time and improved the performance of models. This also allowed for real-time prediction on incoming news articles, ensuring scalability and fast processing times. This approach was crucial for managing real-time data streams from Reddit and News API.

While the project has met its primary objectives, several avenues for further improvement remain. A major drawback of training a deep learning model is the need to acquire a sufficiently large dataset. This issue was partialy addressed in this paper by using a pre-trained large language model. However, employing an LLM introduces its own challenges, such as the requirement for larger storage and more powerful processing units. Techniques like Data Augmentation are available to mitigate this issue, but further research is needed to identify additional methods for training models that achieve acceptable accuracy without relying on a vast dataset.

Further iterations of this system could focus on enhancing the model capabilities. To improve accuracy of our bias detection model, we could explore more advanced architectures, outside of the classifier DL approach, potentially incorporating state of the art language models. Addressing the class imbalance issues, through data augmentation methods could create a more balanced dataset and potentially boost overall performance. Additionally, incorporating topic modeling techniques would provide deeper insights into the themes and subjects covered by different news sources. This could reveal patterns in how various outlets cover specific topics, offering a more nuanced understanding of media bias and sentiment across different subject areas. These enhancements would collectively contribute to a more robust and insightful analysis of news content, providing users with a more comprehensive view of the media landscape.

References

[1] Kocaman, V. (2020, October 25). Text classification in spark NLP with bert and universal sentence encoders. Medium. https://towardsdatascience.com/text-classification-in-spark-nlp-with-bert-and-universal-sentence-encoders-e644d618ca32

[2] Smaller BERT Sentence Embeddings (L-8_H-512_A-8) | sent_small_bert_L8_512 | Spark NLP 2.6.0. (2020, August 25). https://sparknlp.org/2020/08/25/sent_small_bert_L8_512.html

[3] Kokate, A. (2024, May 14). Scraping Reddit and Subreddit Data Using Python and PRAW : A Beginner's Guide | by Archana Kokate|Mar,2024|Medium| https://medium.com/@archanakkokate/scraping-reddit-data-using-python-and-praw-a-beginners-guide-7047962f5d29

[4] Apache-Kafka.(n.d.). https://kafka.apache.org/intro

[5] news_category_*.csv | Dataset | spark-nlp-workshop

[6] Fine-tune Hugging Face models for a single GPU | databricks.com

[7] DistilBERT model by Hugging Face | https://huggingface.co/docs/transformers/en/model_doc/distilbert

[8] CatOps, W. (2024, May 4). ELK Stack: The Essentials of Elasticsearch, Logstash, and Kibana. Medium. https://medium.com/@williamwarley/elk-stack-the-essentials-of-elasticsearch-logstash-and-kibana-3a09ff647352#:~:text=The%20ELK%20Stack%20is%20a,log%20data%20in%20real%20time.

[9] Hube, C., & Fetahu, B. (2018). Detecting biased statements in wikipedia. In Companion proceedings of the the web conference 2018 (pp. 1779–1786).

[10] Gupta, V., Jolly, B. L. K., Kaur, R., & Chakraborty, T. (2019). Clark kent at SemEval-2019 task 4: Stylometric insights into hyperpartisan news detection. In Proceedings of the 13th international workshop on semantic evaluation (pp. 934–938).

[11] Chen, W.-F., Wachsmuth, H., Al Khatib, K., & Stein, B. (2018). Learning to flip the bias of news headlines. In Proceedings of the 11th international conference on natural language generation (pp. 79–88).

[12] Baly, R., Da San Martino, G., Glass, J., & Nakov, P. (2020). We can detect your bias: Predicting the political ideology of news articles. In Proceedings of the 2020 conference on empirical methods in natural language processing (pp. 4982–4991).

[13] Chen, W.-F., Al Khatib, K., Stein, B., & Wachsmuth, H. (2020). Detecting media bias in news articles using Gaussian bias distributions. In Findings of the association for computational linguistics: EMNLP 2020 (pp. 4290–4300).

[14] Krieger, J.-D., Spinde, T., Ruas, T., Kulshrestha, J., & Gipp, B. (2022). A domainadaptive pre-training approach for language bias detection in news. In Proceedings of the 22nd ACM/IEEE joint conference on digital libraries (pp. 1–7).

[15] J. Reis, P. Olmo, F. Benevenuto, H. Kwak, R. Prates, and J. An, Breaking the news: first impressions matter on online news. In ICWSM'15, 2015.

[16] N. Godbole, M. Srinivasaiah, and S. Sekine, Large-scale sentiment analysis for news and blogs. In International Conference on Weblogs and Social Media, Denver, CO, 2007.

[17] K. Alhayyan, and I. Ahmad. Discovering and Analyzing Important Real-Time Trends in Noisy Twitter Streams. Journal of Systemics, Cybernetics and Informatics (JSCI), Vol. 2 – No. 2 – 2017, pp. 25-31

big-data-report.pdf

ORIGINALITY REPORT

15%	11%	9%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Mishra, Pankaj. "Disease Detection in Plants Using UAS and Deep Neural Networks", Tennessee State University, 2024 Publication	1%
2	aclanthology.org Internet Source	1%
3	Amir Shachar. "Introduction to Algogens", Open Science Framework, 2024 Publication	1%
4	peer.asee.org Internet Source	1%
5	Francisco-Javier Rodrigo-Ginés, Jorge Carrillo-de-Albornoz, Laura Plaza. "A systematic review on media bias detection: What is media bias, how it is expressed, and how to detect it", Expert Systems with Applications, 2023 Publication	1%
6	xbiudp.1a-webverzeichnis.de Internet Source	1%

7	Submitted to AlHussein Technical University Student Paper	<1 %
8	Submitted to Coventry University Student Paper	<1 %
9	Submitted to University of Limerick Student Paper	<1 %
10	assets.researchsquare.com Internet Source	<1 %
11	digitalscholarship.unlv.edu Internet Source	<1 %
12	origin.geeksforgeeks.org Internet Source	<1 %
13	www.pitsolutions.ch Internet Source	<1 %
14	1library.net Internet Source	<1 %
15	Submitted to Technological University Dublin Student Paper	<1 %
16	Submitted to University of Salford Student Paper	<1 %
17	dokumen.pub Internet Source	<1 %
18	riunet.upv.es Internet Source	<1 %

19	rua.ua.es Internet Source	<1 %
20	cris.unibo.it Internet Source	<1 %
21	iejrd.com Internet Source	<1 %
22	"Natural Language Processing and Information Systems", Springer Science and Business Media LLC, 2023 Publication	<1 %
23	Submitted to Universiteit van Amsterdam Student Paper	<1 %
24	Submitted to University of Wollongong Student Paper	<1 %
25	Submitted to Liverpool John Moores University Student Paper	<1 %
26	Submitted to University of Westminster Student Paper	<1 %
27	Submitted to North Central State College Student Paper	<1 %
28	Submitted to University of Northumbria at Newcastle Student Paper	<1 %
29	ntnuopen.ntnu.no	

Internet Source <1 %

30 www.stackchief.com
Internet Source <1 %

31 2pforinterview.blogspot.com
Internet Source <1 %

32 acikerisim.aydin.edu.tr
Internet Source <1 %

33 arxiv.org
Internet Source <1 %

34 "Bildverarbeitung für die Medizin 2018", Springer Nature, 2018
Publication <1 %

35 123dok.net
Internet Source <1 %

36 Tien Quang Dam, Nghia Thinh Nguyen, Trung Viet Le, Tran Duc Le, Sylvestre Uwizeyemungu, Thang Le-Dinh. "Visualizing Portable Executable Headers for Ransomware Detection: A Deep Learning-Based Approach", JUCS - Journal of Universal Computer Science, 2024
Publication <1 %

37 www.intechopen.com
Internet Source <1 %